WOWvx SettingsAPI

User Manual

Philips 3D Solutions

Document Information

Info	Content
Title	WOWvx SettingsAPI, User Manual
Date	31 August 2007
Security	The attached material and the information contained herein are proprietary to Philips 3D Solutions. Copying, reproduction, adaptation, modification or dissemination in whole or part is not permitted without written permission from Philips 3D Solutions.
Contact address	Philips 3D Solutions High Tech Campus 27 5656 AE Eindhoven The Netherlands, E-mail support: 3DService@philips.com E-mail general: 3DSolutions@philips.com Website: www.philips.com/3dsolutions

Table of Contents

1 Introduction

The WOWvx SettingsAPI is a library of functions that help application programmers to monitor and control WOWvx 3D displays. The library implements three types of functions:

1) Functions that describe the 3D content. Traditionally, the content is described by a header in the top left corner of the display (see the '3D Interface Specification – white paper' for more details on this header). The WOWvx SettingsAPI provides an alternative if drawing a header is impractical for any reason.
2) Functions that switch the display into or out of the standby mode. Standby mode enables significant power savings if the display is unused, but still powered on.
3) Functions that change the user display settings. The user display settings allow a user to change parameters to suit his or her preferences. This functionality is also provided by the Display Control Tool, a utility that is delivered together with the Philips WOWvx 3D displays.

The API is implemented in the form of a dynamic link library (.dll). It is targeted towards C/C++ applications written in Microsoft Visual Studio 6.0 or Microsoft Visual Studio 2005. It may be possible to use the API from other environments and languages, but this is not formally supported.

In the rest of the document, we assume that you are familiar with the Microsoft Visual Studio development environment and C/C++. We also assume that you are familiar with WOWvx 3D displays, most notably that you have read the '3D Interface Specification – white paper' and the 'Display Control Tool – user manual'. The former can be downloaded from the 3D Solutions web site, while the latter is installed on your PC when the Display Tools, which are delivered with every Philips WOWvx 3D display, are installed.

1.1 Document contents

Chapter 2 describes the files that are distributed with the WOWvx SettingsAPI.

Chapter 3 explains some basic concepts from the WOWvx SettingsAPI and contains a general description of the API.

Chapter 4 explains the example application.

Chapter 5 contains the troubleshooting guide, including the known problems.

Note that there are two more files that contain relevant information:

- inc\WOWvxSettingsAPI.h: this file contains the API reference/specification.

- inc\ReleaseNotes.h: this file contains the release notes, including some limitations of the current implementation.

Both files are in the form of documented C header files. This documentation has been converted to html files that are easier to read with Doxygen (http://www.doxygen.org, but you don't need to know about this tool in order to read the documentation). The main page is doc\html\index.html.

2 Contents of the WOWvx SettingsAPI

The WOWvx SettingsAPI consists of the following files:

inc\WOWvxSettingsAPI.h	The header file that contains the description of the WOWvx SettingsAPI. This header file has been extensively documented and the entire API specification can be found here.
inc\ReleaseNotes.h	This header file contains no C/C++ definitions. It contains the release notes of the WOWvx SettingsAPI in a comment.
doc\WOWvxSettingsAPI User Manual.pdf	This document.
doc\html\*.*	This directory contains the Doxygen output for the previous two files. The file index.html is the main file.
lib\WOWvxSettingsAPI.dll	The Dynamic Link Library containing the object code of the API.
lib\WOWvxSettingsAPI.lib	The library that makes linking to the .dll file possible.
WOWvxSettingsExample\PreviewPane.cpp	Example code for a preview window that puts some 3D content (without content header) on the 3D display.
WOWvxSettingsExample\PreviewPane.h	
WOWvxSettingsExample\resource.h	Header file containing resource constants.
WOWvxSettingsExample\StdAfx.cpp	Visual Studio files containing headers used throughout the example project.
WOWvxSettingsExample\StdAfx.h	
WOWvxSettingsExample\WOWvxSettingsExample.clw	Visual Studio Class Wizard file.
WOWvxSettingsExample\WOWvxSettingsExample.cpp	Example code for the main application. This is a standard MFC application that contains almost no specific code.
WOWvxSettingsExample\WOWvxSettingsExample.h	
WOWvxSettingsExample\WOWvxSettingsExample.dsp	Microsoft Visual Studio 6.0 project files for the example application.
WOWvxSettingsExample\WOWvxSettingsExample.dsw	
WOWvxSettingsExample\WOWvxSettingsExample.sln	Microsoft Visual Studio 2005 project files for the example application.
WOWvxSettingsExample\WOWvxSettingsExample.vcproj	
WOWvxSettingsExample\WOWvxSettingsExampleDlg.cpp	Code for the dialog window of the example application. This is where the API is actually used.
WOWvxSettingsExample\WOWvxSettingsExampleDlg.h	
WOWvxSettingsExample\res\WOWvxSettingsExample.ico	The example application icon.
WOWvxSettingsExample\res\WOWvxSettingsExample.rc2	Additional application resources (not used).

WOWvxSettingsExample\res\Hakim.bmp	The image used as example content.

3 Using the WOWvx SettingsAPI

3.1 General

The WOWvx SettingsAPI supports multiple WOWvx 3D displays attached to the same PC. It is able to control them independently. It knows to which display a command must be sent via the WOWvxSettingsHandle, which is linked to a 3D display.

All functions return an error code that is 0 on success.

3.2 Initializing

You can create a WOWvxSettingsHandle with the WOWvxSettingsInit function. This function takes the 3D display number and returns a handle to it. It will also report how many 3D displays are connected to the system. If this number is not equal to the number of displays you have physically connected to the system, it can be that Windows has not recognized the 3D display, e.g. in case you connected the display without rebooting afterwards.

If you connect a display through a DVI splitter, communication may not be possible, depending on the splitter. Please contact your 3D display vendor if you want to connect multiple 3D displays through a DVI splitter.

You can connect to a different display by calling the WOWvxSettingsInit function with a different display number. It is also possible, though not recommended, to connect to the same display, but via a different handle. For standby and user settings functions, this will work as expected (the last call 'wins'). However, for the content functions this will give unexpected results. This is because the content functions expect that they are the only ones setting the content parameters. This is not unlike setting those parameters with the content header: you can only draw one header. Note that the WOWvx SettingsAPI will behave the same if one application gets two handles to the same 3D display as if two applications get handles to the same 3D display.

When you are done with a WOWvxSettingsHandle, you should release it with the WOWvxSettingsExit function.

3.3 Get and Set functions

For all display settings a Set and a Get function are available. The Set function will change the setting in the display. The Get function will retrieve the actual setting in the display. Note that if different applications modify the settings (e.g. with the Display Control Tool), the value you retrieve with a Get function may be different from the value you previously set with the Set function.

All integer parameters of the Set functions can be passed the value of WOW_VX_SETTING_DEFAULT to set the default value. However, often the default value is a special case and not a specific value. For instance, a default value for WOWvxUserSetVisualization means that the value specified by the content is used. In other cases the default value is the factory default of the display: e.g. a default value for WOWvxUserSetBrightness means a value of 128. A call to WOWvxUserGetBrightness will then return 128. Whenever a default value resolves to a specific value, this value will be noted in the function description.

All Set functions will change the setting on the display immediately. However, no guarantees are made about the speed of the communication. Specifically, it is not possible to draw a new bitmap and update the content settings so the correct settings are in effect at the moment the bitmap is drawn. There is always a chance that the settings will be in effect one frame too early or too late. You must use the content header if you need frame-accuracy for the content settings. Note that this is most visible at the moment that the display is switched from or to 3D mode. You can use a completely black frame at the moment of the switch to hide this from the viewer.

Values that you change with the WOWvx SettingsAPI will be lost as soon as the 3D display is switched off. You can save the user settings (not the content settings or standby settings) with the WOWvxUserSaveSettings function. Note that this only works if the Display Tools that are delivered with every Philips WOWvx 3D display are installed.

3.4 Standby mode

In standby mode, the display backlight is switched off, which greatly reduces power consumption. However, you should be careful when using this function. Apart from disabling the video signal (i.e. unplugging it), the WOWvx SettingsAPI is currently the only way to switch a display to standby mode. This means that if an application switches the display to standby and then crashes, Windows has no way to switch the display back to the normal mode. The only way to restore the display in that case is to manually power it off and switch it on again. The 3D display will not remember that it was in standby mode before power off.

3.5 Priorities of competing settings

There are two ways to switch a WOWvx display to 3D mode: using the SettingsAPI or displaying content that has a header in the top left corner. The SettingsAPI will always have priority. When the SettingsAPI is used to switch the display to 3D mode, any content header that may be present is completely ignored, including any settings that the header may contain. When the SettingsAPI sets the display to 2D mode, a content header will switch the display to 3D mode. In that case, the get functions of the SettingsAPI will *not* return the values as they are set by the content header.

Some settings are available in both a content variant and a user variant. These settings are: factor, screen depth, clear edge, and visualization.

How the API handles these settings is documented in the API specification, but the short version is: factor and screen depth are applied both, while for clear edge and visualization, the user variant overrides the content variant.

4 The WOWvxSettingsExample example application

4.1 Introduction

The WOWvxSettingsExample is a very simple MFC (Microsoft Foundation Classes) application that provides functionality similar to the Display Control Tool, which is delivered with every Philips WOWvx 3D display. However, the Display Control Tool can only set the user settings, while the WOWvxSettingsExample can also set the content type and standby mode (on exit the example restores the original configuration).

The example displays some 3D content in a preview window. This window is shown on the 3D display. If the 3D display is the only display connected to your PC, this means that both the example dialog and the 3D content are on the 3D display. This can have some strange looking effects if the display is switched to 3D.

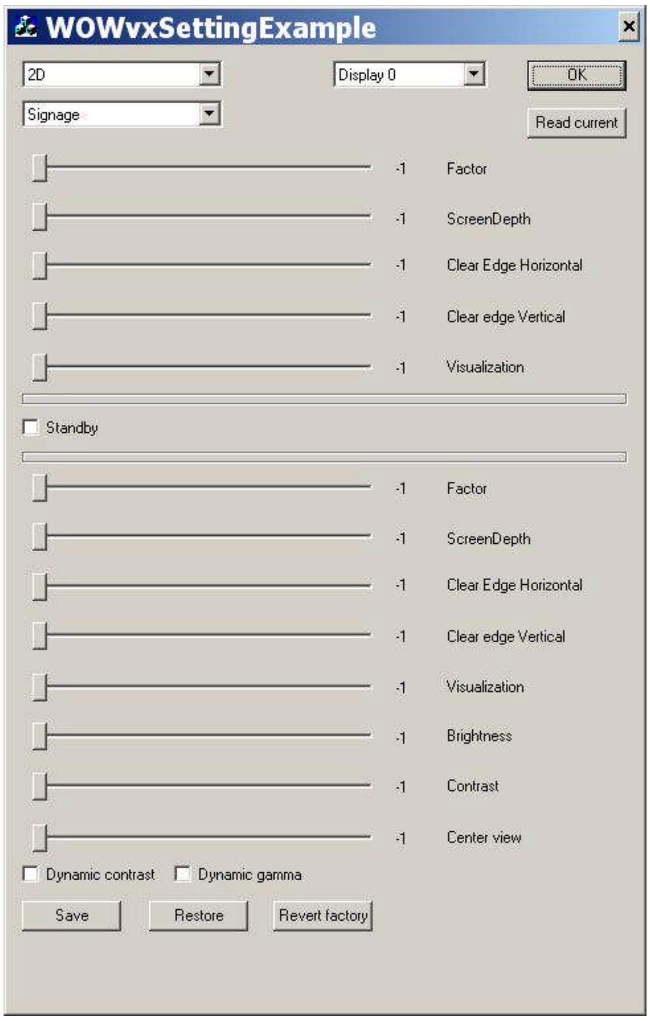
The preview content must be shown in the 1920x1080 resolution. This is due to the specification of the Declipse format (see the 3D Interface Specification white paper), which enforces the interleaving of content. The example will not work if you have a 20" 3D display or if your display is set at the wrong resolution. In that case, you should modify the example application such that it shows some different 3D content.

4.2 The example dialog

The example will start with the dialog depicted in Figure 1. It is divided in three parts. At the top is the content section. In the middle is the standby section. At the bottom is the user settings section.

Most controls work with a slider bar, where the slider setting (also displayed next to the slider) corresponds with the integer parameter of the WOWvx SettingsAPI function that is associated with the control. The special value -1 is mapped to the default value: WOW_VX_SETTING_DEFAULT.

For the functions where a slider is not convenient, other controls such as drop-down boxes, checkboxes, or buttons are used.



2D
Signage
Display 0

-1 Factor
-1 ScreenDepth
-1 Clear Edge Horizontal
-1 Clear edge Vertical
-1 Visualization

-1 Factor
-1 ScreenDepth
-1 Clear Edge Horizontal
-1 Clear edge Vertical
-1 Visualization
-1 Brightness
-1 Contrast
-1 Center view

Figure 1. The example dialog box

At the very top in the middle is a drop-down box that determines which 3D display is controlled. In case you have multiple 3D displays connected to your system, you can choose which one to control via this list.

At the very top left is a drop-down box that determines the display mode. As soon as that is changed to a 3D mode, the 3D display will display the content in 3D. The drop-down box below it determines the content type, which influences the value of some of the default settings of the 3D rendering parameters (see below).

The five sliders below the content type are 3D rendering parameters. You can change them and observe the effect that they have on the displayed content. Note that these parameters are in effect only if the display is switched to 3D mode by the WOWvx SettingsAPI. They have no effect if the display is switched to 3D mode by a header in the content.

The standby mode is controlled via a simple checkbox. Checking the box switches the display in standby, un-checking it switches it back to the normal mode. Note that if you only have a 3D display connected, un-checking may be difficult because the display is black. Pressing Alt-F4 (exit application) can help in that case. Manually switching off and on the display will also help.

The bottom part of the dialog box contains the user settings, which should be familiar to anybody who has used the Display Control Tool. Note that the ranges presented to the user by the Display Control Tool are not the same ranges provided in this example. The Display Control Tool has simpler ranges and does not allow you to do senseless things like setting the contrast to 0 (which results in a black screen).

Note that both the content section and the user section contain the same parameters. The content settings belong to specific content. If an application uses the WOWvx SettingsAPI to display 3D content, it should use these settings. The user settings are used by the user to change parameters for all content, based on his or her preferences. Your application should only change them if it performs more or less the same function as the Display Control Tool.

4.3 How the code works

The WOWvxSettingsExample is a straightforward MFC dialog-based application. Because of that, most functionality is implemented in the CWOWvxSettingsExampleDlg class. Additionally, CWOWvxSettingsExampleApp::InitInstance() creates an instance of a CPreviewPane class. This CPreviewPane is responsible for drawing 3D content. However, as this content is already available as a resource (the file res\Hakim.bmp), the code of CPreviewPane is trivial.

The code of CWOWvxSettingsExampleDlg is also easy to understand. Linking the controls to variables is handled by MFC. Call-backs on relevant events (such as when a slider is moved) are also handled by MFC. The code that links these events to calls to the WOWvx SettingsAPI is as short as a few lines. The comments available in WOWvxSettingsExampleDlg.cpp should be sufficient to understand how the API can be used.

5 Troubleshooting

Problem	**Possible solution**
The call to WOWvxSettingsInit fails, even though a 3D display is connected to the PC.	First check if the Display Control Tool can communicate with the 3D display. If not, first resolve this problem (e.g. a reboot may be required after connecting a display, only specific video cards are supported, a DVI splitter may not work).
Compiled example code 'disappears' under Windows Vista	If the API is installed in its default location, under 'Program Files', Windows Vista will put the compiled files at some other location (this is Vista's virtualization feature). It is better to copy the example code somewhere else.

- 0 – 0 – 0 – 0 – 0 -